Optimizing communications on clusters of multicores

Alexandre DENIS

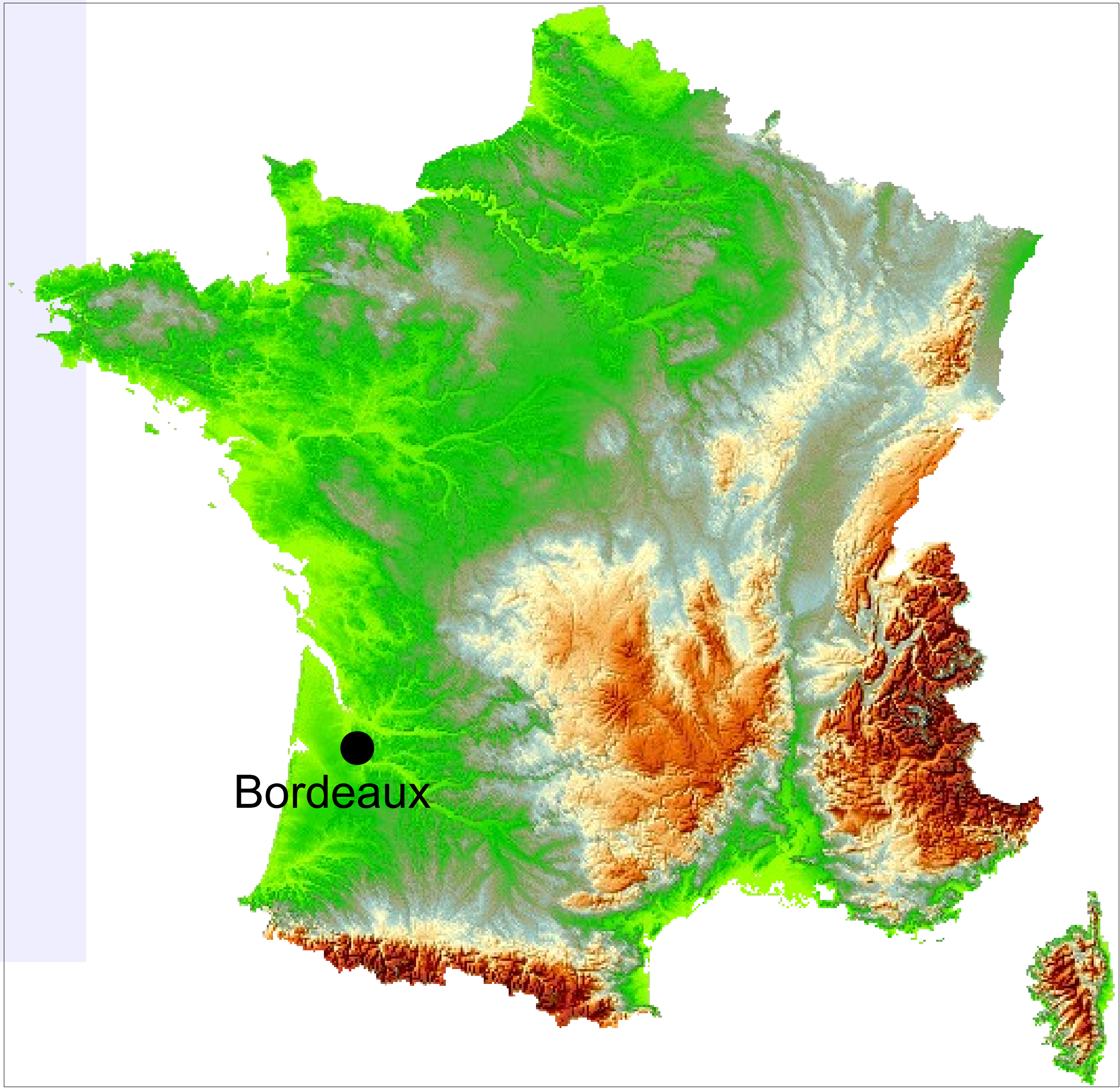
with contributions from:
Elisabeth Brunet, Brice Goglin, Guillaume Mercier, François Trahay

Runtime Project-team
INRIA Bordeaux - Sud-Ouest

INRIA-Illinois workshop
Paris, June 2009

The RUNTIME Team

- Mid-size research group
  - Head: Raymond Namyst
  - 6 permanent researchers
  - 2 engineers
  - 6 PhD students
- Part of
  - INRIA Bordeaux Sud-Ouest Research Center
  - Computer Science Lab at University of Bordeaux 1 (LaBRI)



Current trends

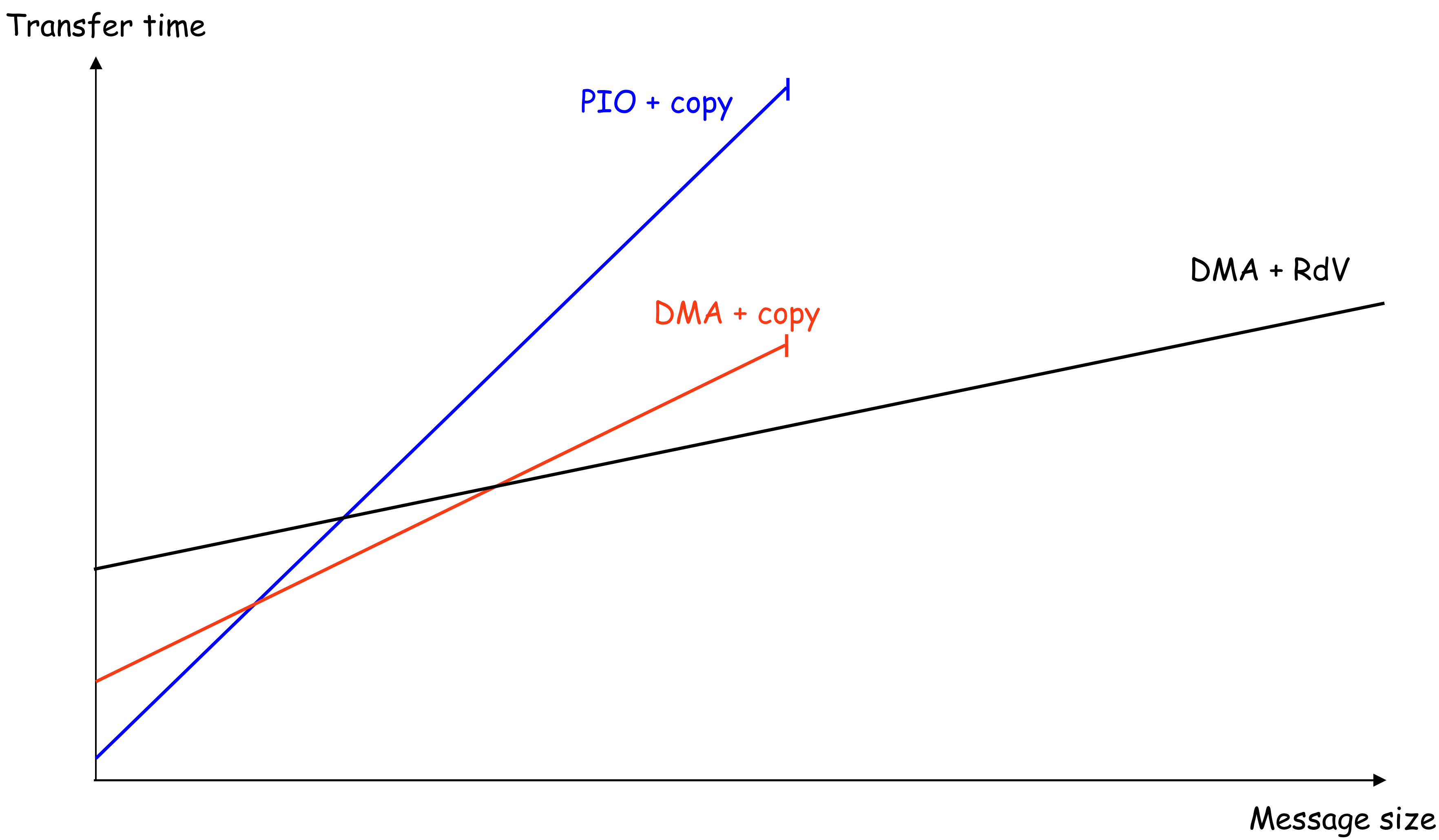
- The world is goig multicore
  - Currently: 4-8-16 cores per node
  - Tomorrow: hundreds of cores per node
  - Sometimes: multiple NIC per node
- New programming models
  - “Pure-MPI” approach
    - One MPI process per core
  - **Hybrid** approach (MPI + threads/OpenMP)
    - One MPI process per node/socket

A thread-aware approach for communications

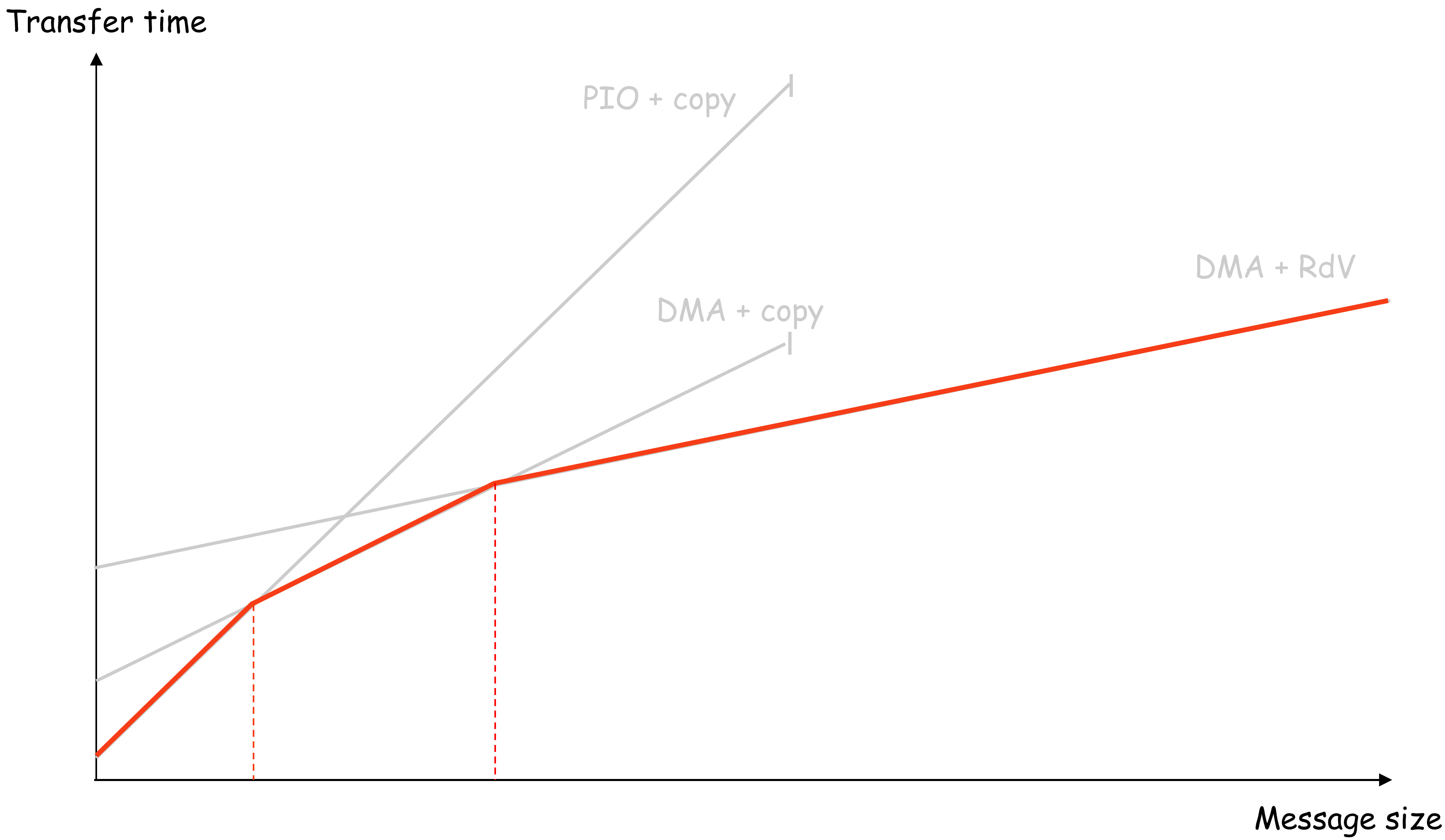
- Impact of multi-core on communications
  - Communication library must support multi-threading
  - Multiple flows from multiple threads share a NIC
  - ➔ **Decouple** `MPI_Send/Recv` from NIC send/recv

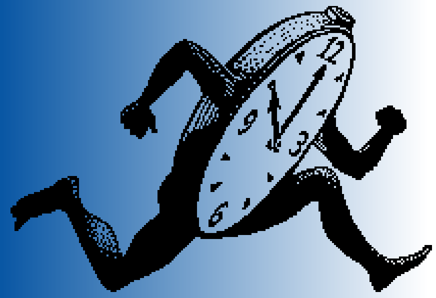
- New opportunities for optimization!
  - Aggregate packets from different threads
  - The “best” optimization strategy depends on:
    - the capabilities and performance of the underlying network
    - host architecture
    - applications requirements
  - Opportunistically use idle cores
  - ➔ **Decide optimization at run time**

Implementing data transfers efficiently

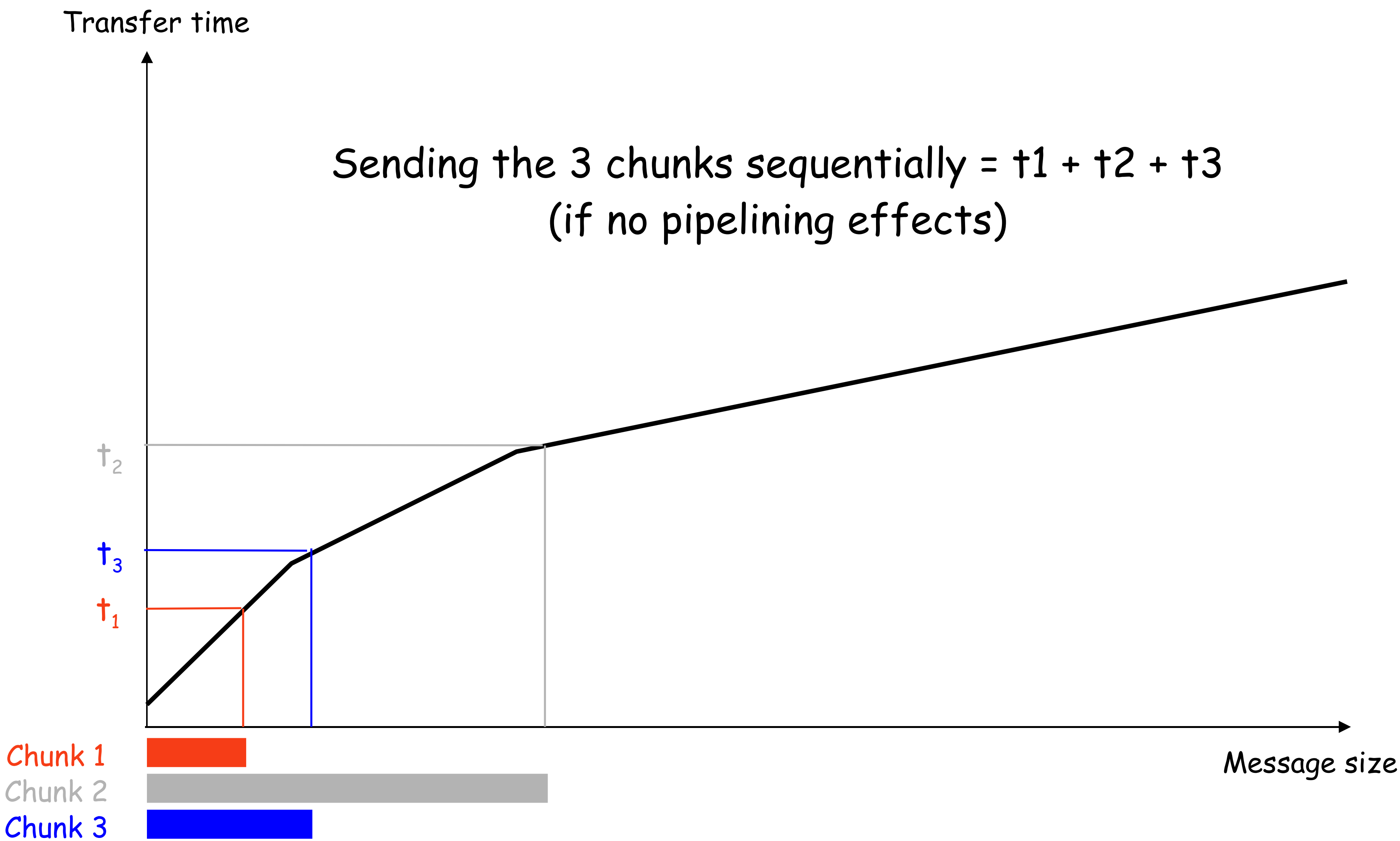


Implementing data transfers efficiently





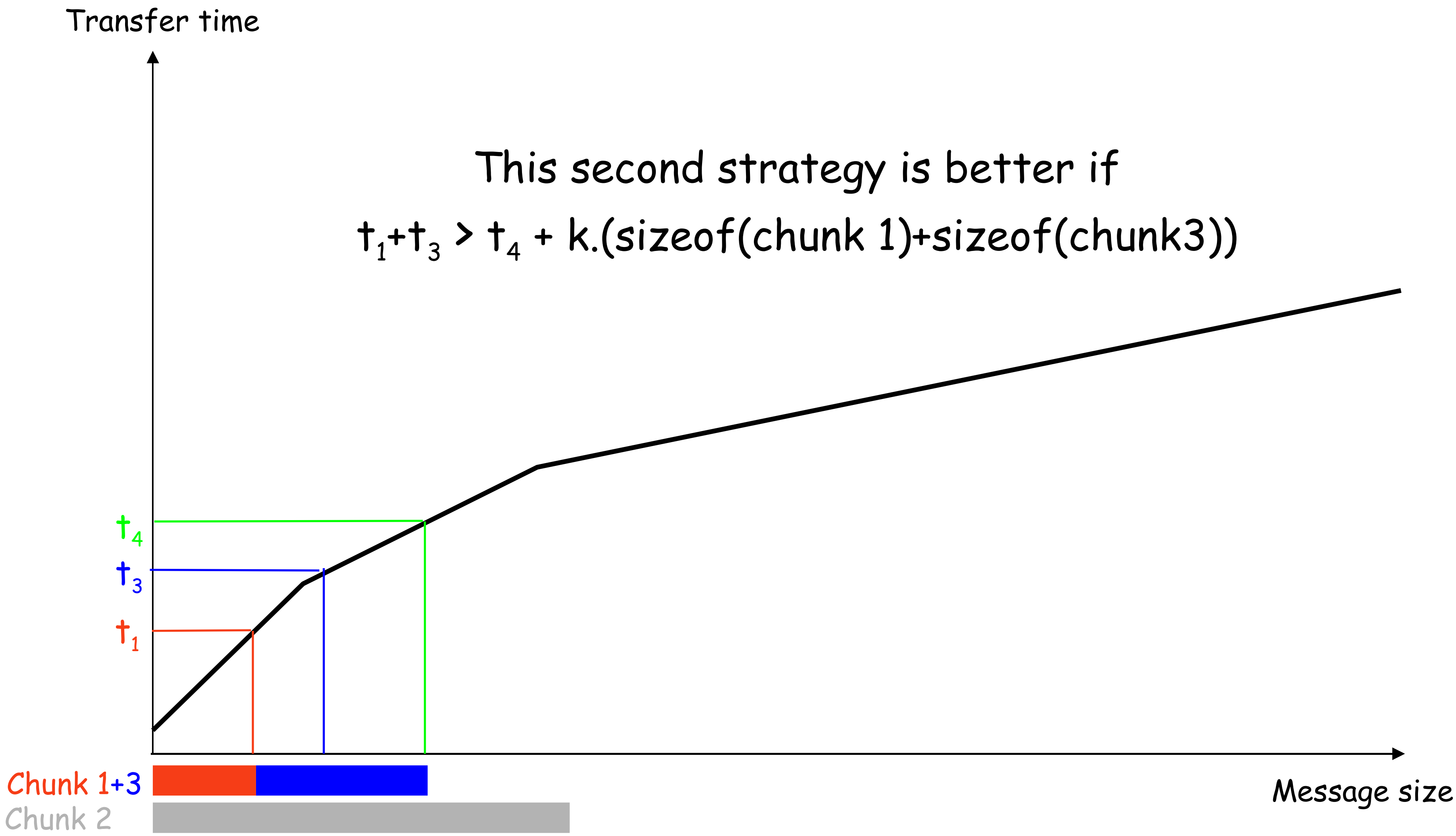
Implementing data transfers efficiently



Implementing data transfers efficiently

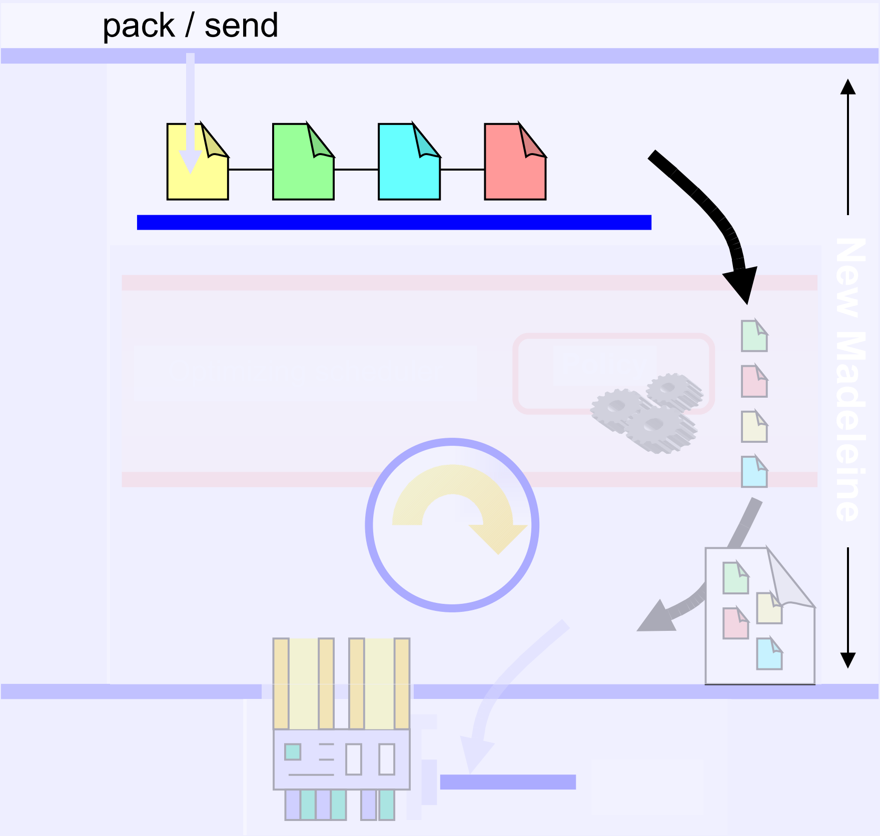
This second strategy is better if

$$t_1 + t_3 > t_4 + k.(\text{sizeof}(\text{chunk 1}) + \text{sizeof}(\text{chunk3}))$$



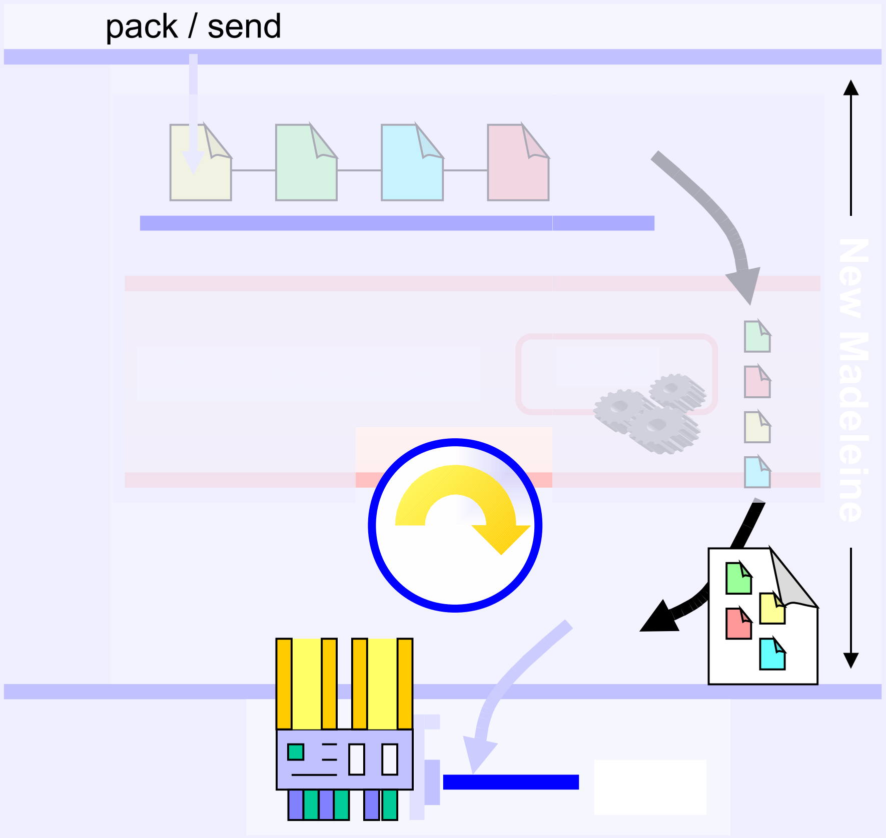
NewMadeleine architecture: the interface layer

- Application submits requests
  - Non-blocking operation
- Meta-data are associated to data
  - Reordering constraints
  - Tag, seq number, etc
- User communication interfaces
  - Incremental message building interface
  - Send-receive interface
  - Simple MPI implementation: Mad-MPI



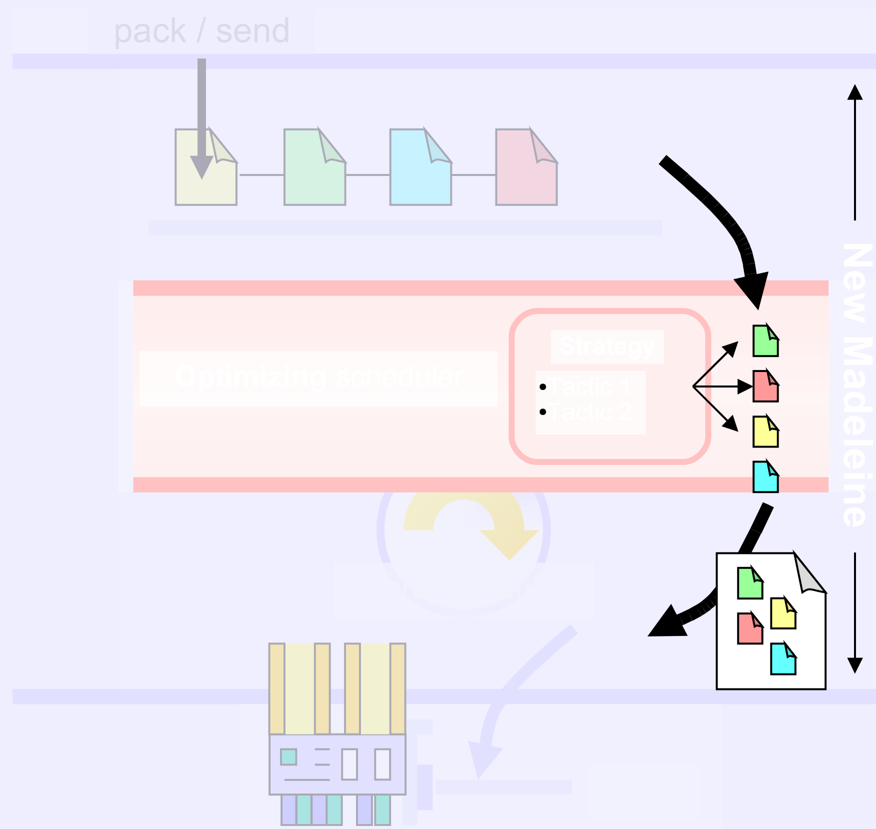
NewMadeleine architecture: the network layer

- NewMadeleine activity triggered by NIC
  - Busy NIC → gather application requests
  - Idle NIC → invoke the scheduler
    - Analysis of the user request backlog
    - Application of a strategy
    - Synthesis of a network request
- Available drivers
  - Myrinet MX, Infiniband verbs, Quadrics Elan, SCI Sisci, TCP sockets



NewMadeleine architecture: strategies

- Strategy invoked when NIC becomes idle
- Combined with any network(s)



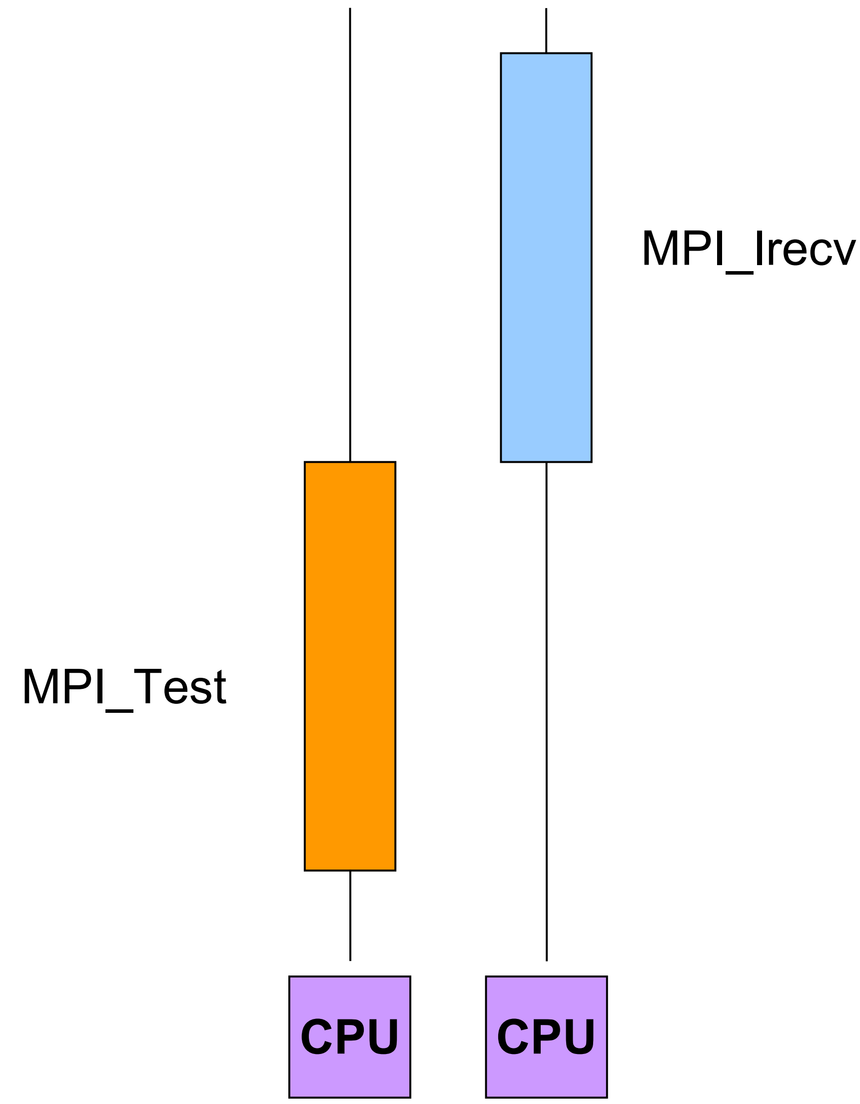
- Currently available:
  - Default
    - Raw transmission
  - Aggregation
    - Aggressive aggregation of small packets
  - Multirail
    - Split packets over multiple networks
  - QoS
    - Priority-based scheduling

Mixing network and threads

- Multiple levels of threads support in communication subsystem
- Level 1: Thread safety
  - Allow simultaneous access to the library
- Level 2: Background progression of communication
  - Make communications progress in background (rendez-vous, non-blocking)
- Level 3: Parallel processing
  - Use several cores to process communication

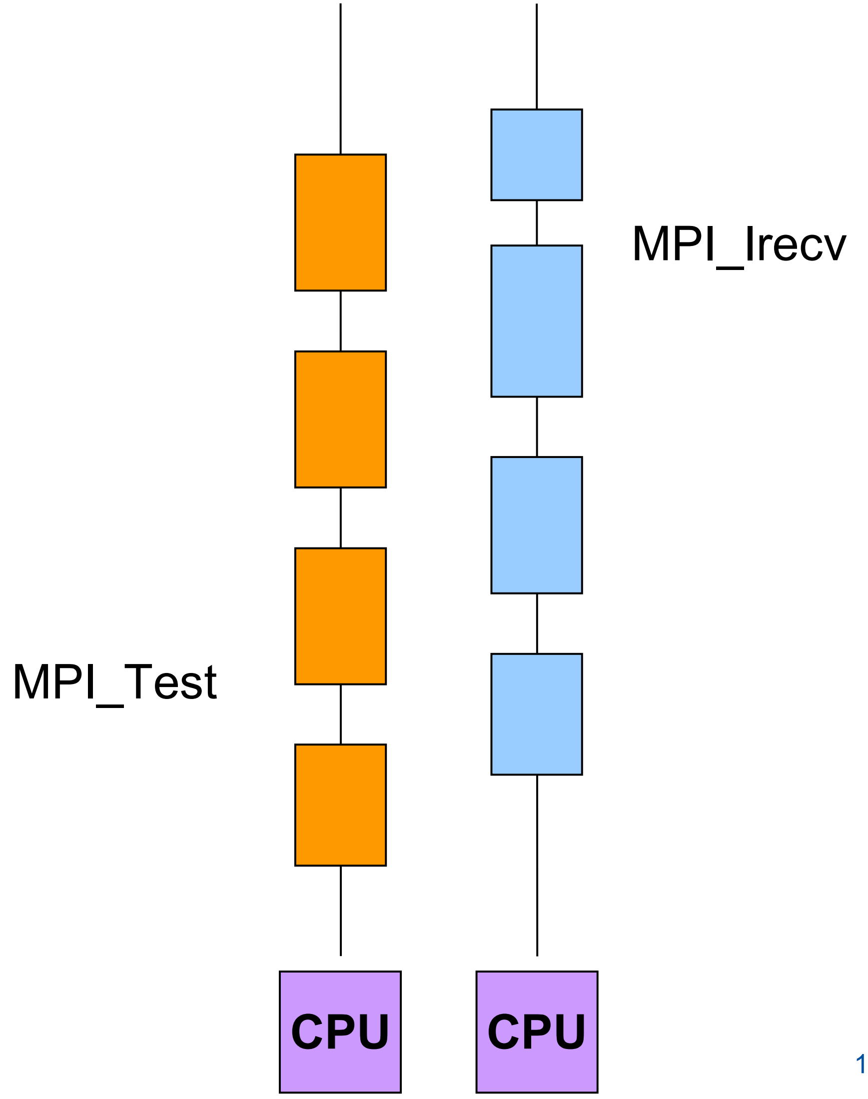
Ensuring thread-safety

- Level 1: thread safety
  - Coarse grain
    - Library-wide mutex
    - Avoids simultaneous access to the library
    - Overhead = 140ns
      - negligible



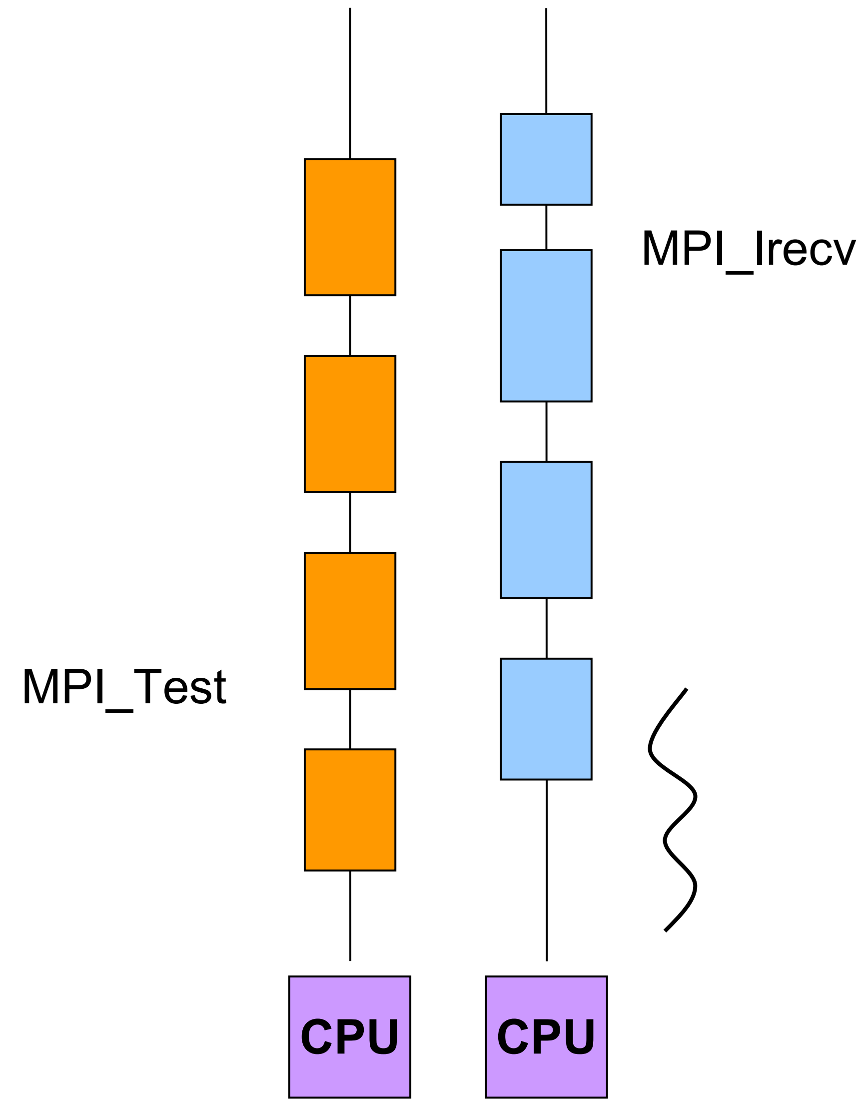
Ensuring thread-safety

- Level 1: thread safety
  - Coarse grain
    - Library-wide mutex
    - Avoids simultaneous access to the library
    - Overhead = 140ns
  - Fine grain
    - Action-wide mutexes
    - Local thread-safety
    - Allows simultaneous access to the library
    - Overhead = 230ns



Background processing

- Level 2: background processing
- A progression *thread* per NIC
  - Rendezvous handshake progression
    - MX/Myrinet, OpenMPI/TCP
  - Priority issue on overloaded systems
  - Overhead :
    - 400ns (inter-core, same chip synchronization)
    - 2-3us. (inter-chip synchronization)
    - Depends on thread placement

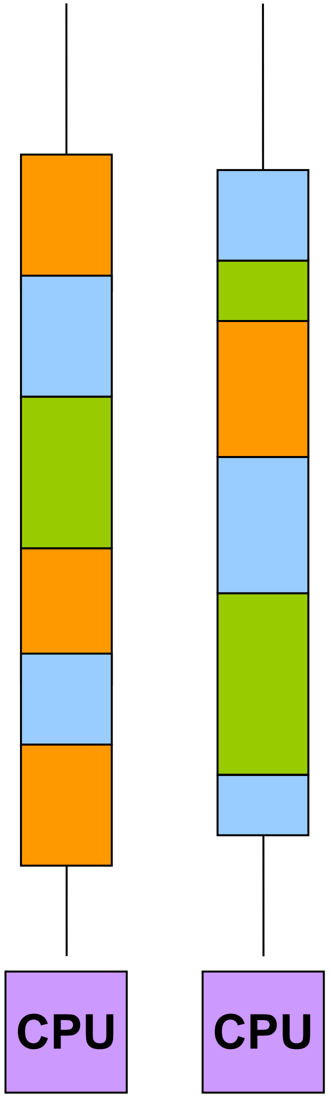


Parallel processing of communication flows

- Level 3: take benefit from multi-core:

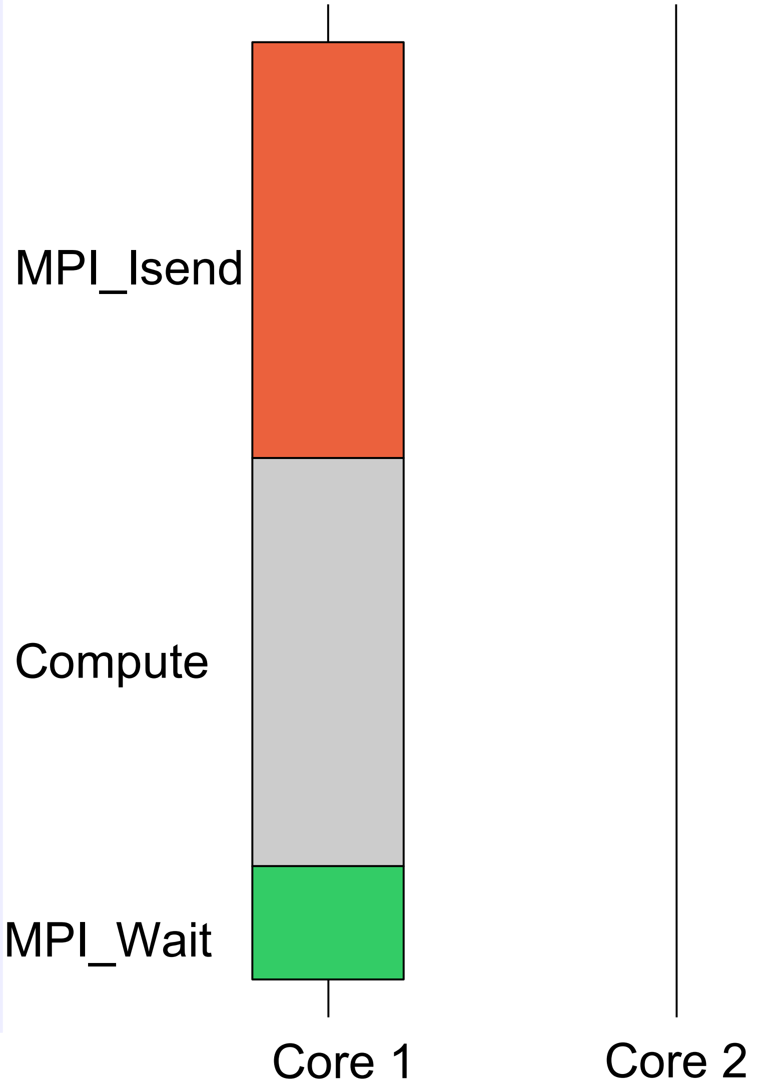
The **PIOMan** communication manager

- Communication processing seen as a sequence of operations (*tasklets*)
  - Operations may be scheduled on any core
    - through *hooks* in thread scheduler: idle, timer, context switch
  - Load balancing of processing
    - Idle cores 'help' working cores
  - Offloading of asynchronous operations
  - No priority issue
  - Reduced overhead : 400ns
    - Placement is controlled
    - Less context switches



Offloading small messages

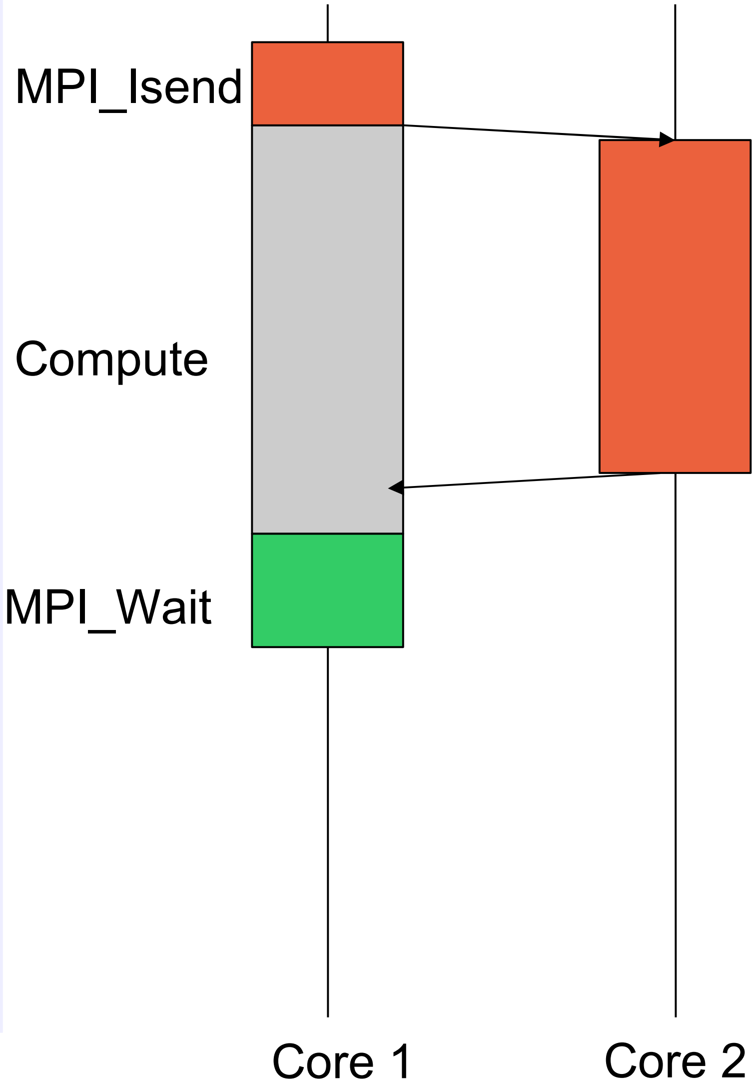
- Sending small messages consumes CPU
  - `memcopy` or PIO may monopolyze a CPU for dozens of μs
- Even a MPI_Isend can be split
  - Split the non-blocking send into basic operations
    - a) Register the MPI request
    - b) Submit the packet to the NIC
  - Spread the operations on cores

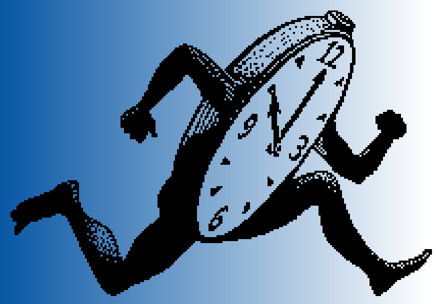


Offloading small messages

- Sending small messages consumes CPU
  - `memcopy` or PIO may monopolyze a CPU for dozens of μs

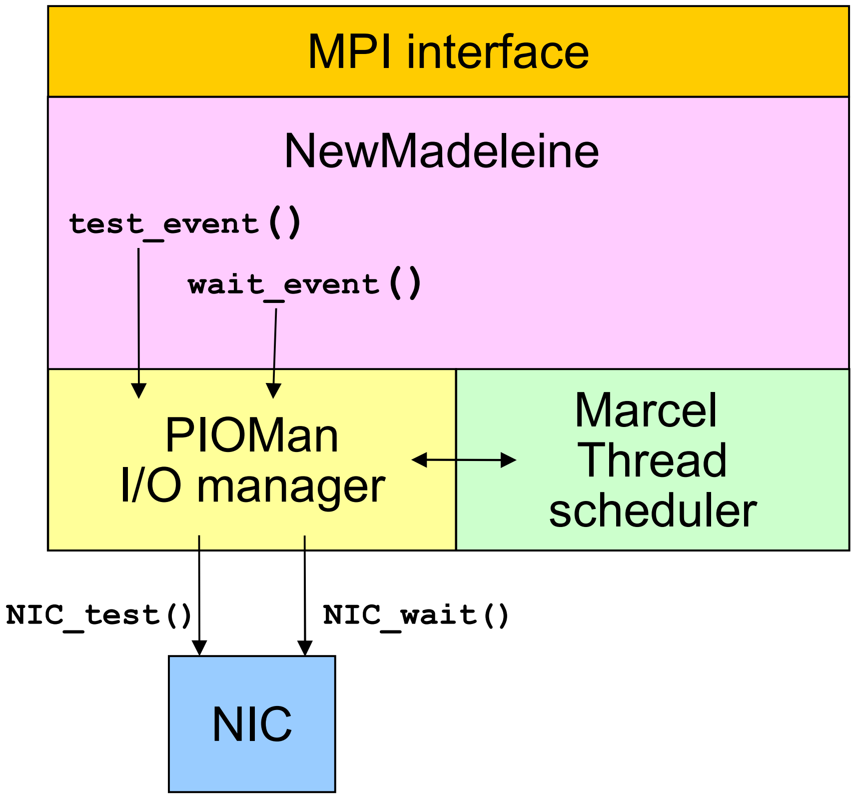
- Even a MPI_Isend can be split
  - Split the non-blocking send into basic operations
    a) Register the MPI request
    b) Submit the packet to the NIC
  - Spread the operations on cores

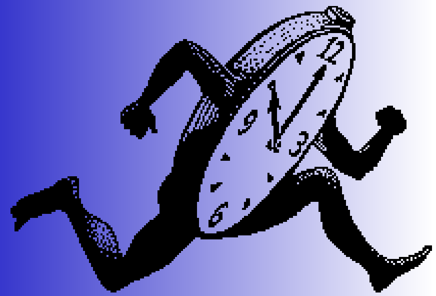




The PIOMan communication engine

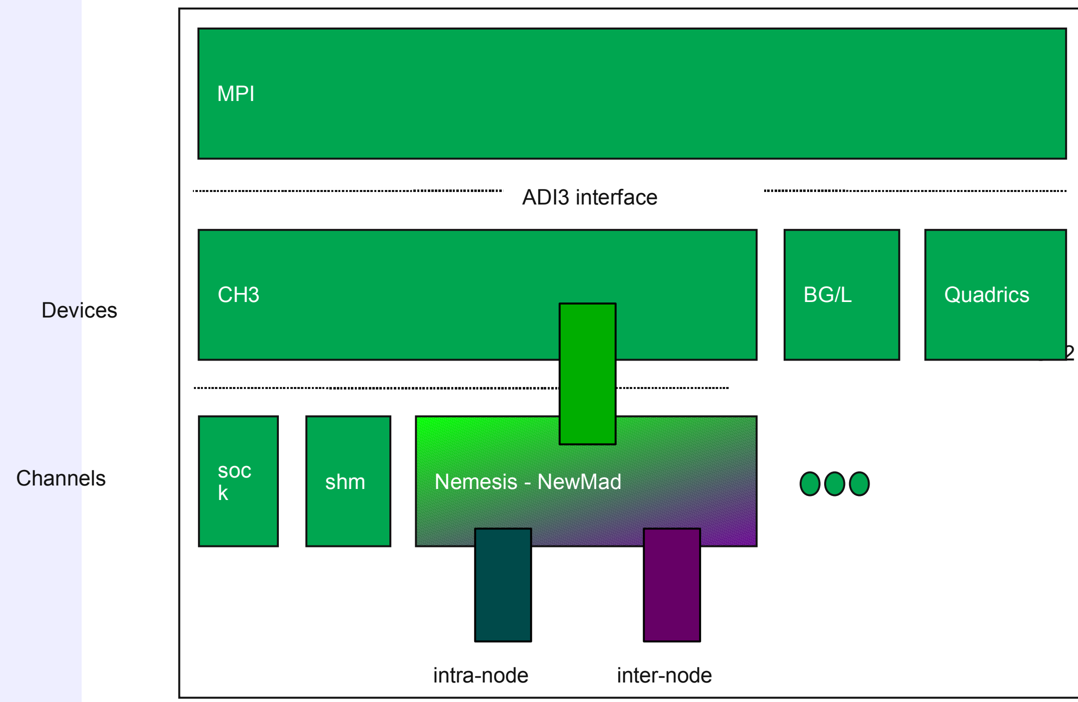
- PIOMan: the PM2 I/O event manager
  - Thread-aware I/O event manager
    - Offload message submissions on idle cores, if available
    - Uses interrupt and/or polling transparently, depending on system load
    - Makes communication progress asynchronously
      - No thread needed in application
      - No priority issue, even on overloaded systems
  - Well integrated with the Marcel thread scheduler





Bridging the gap with standard API

- Mad-MPI: a light MPI implementation on top of NewMadeleine
  - Bringing the performance of NewMadeleine to simple MPI applications
- NEMESIS/NewMadeleine: a new architecture for MPICH2
  - Towards a powerful optimization engine for MPI



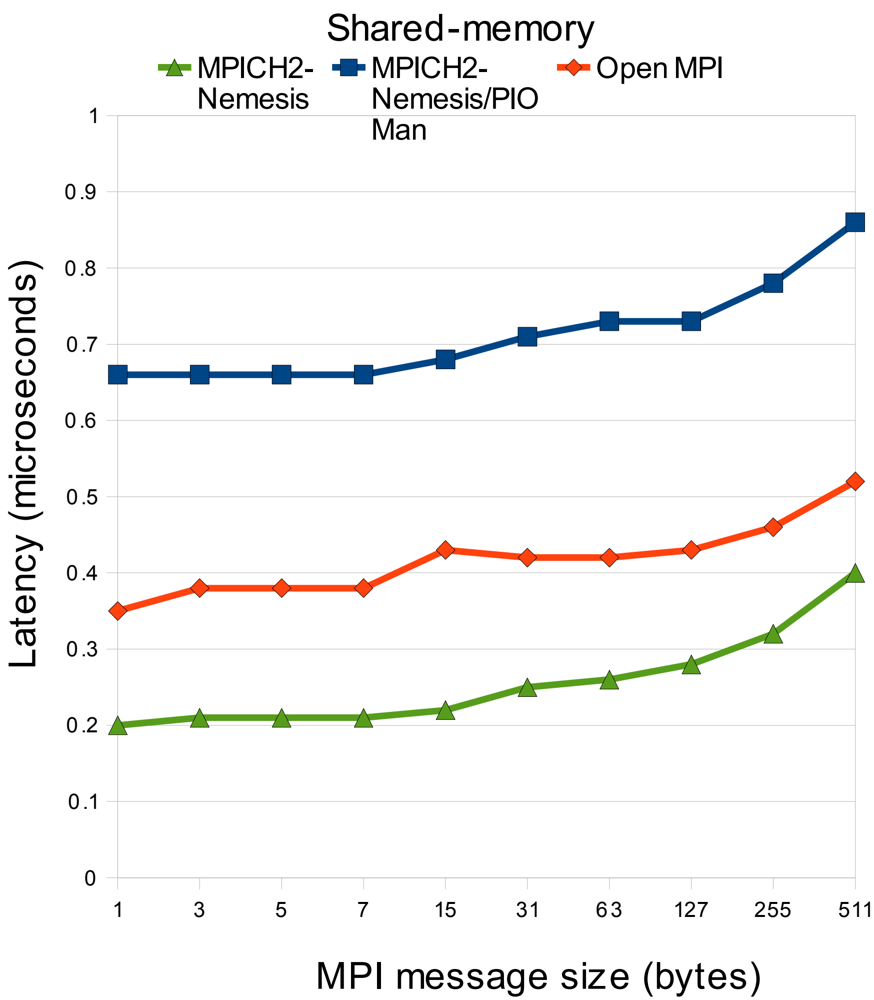
MPICH2 over NewMadeleine

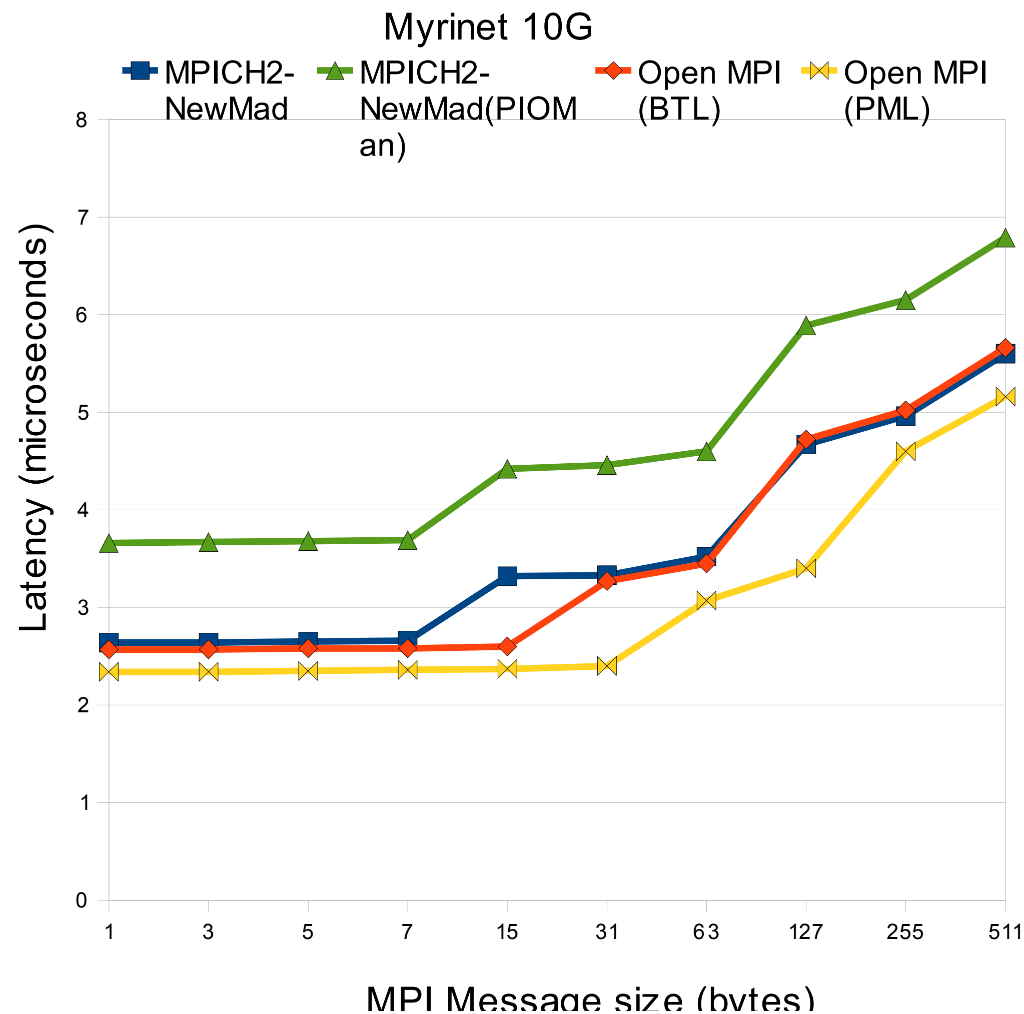
- The preliminary version was straightforward to implement
  - MPI_xxx point to point communication are (almost) directly mapped on nmad_xxx
  - The NEMESIS is used for intra-node (shared memory) communication (joint work with Argonne NL)
  - Published at IPDPS 2009
- Performance is good
  - Low latency overhead, bandwidth is the same
- Scientific challenges
  - Optimization of MPI datatypes management
  - Support for collective operations within NewMadeleine
  - Full support of dynamicity over heterogeneous (or multi-rails) configurations

Experimental testbed

- For point-to-point experiments
  - Two 2-quadcore nodes with Intel Xeon 3.16GHz
  - 4GB of memory per node
  - One Myrinet 10G NIC and one ConnectX Infiniband HCA
- For NAS parallel benchmarks
  - Ten 4-dualcore nodes with AMD Opteron 2.6GHz
  - 32 GB of memory per node
  - One Infiniband 10G HCA

PIOMan's impact on latency performance

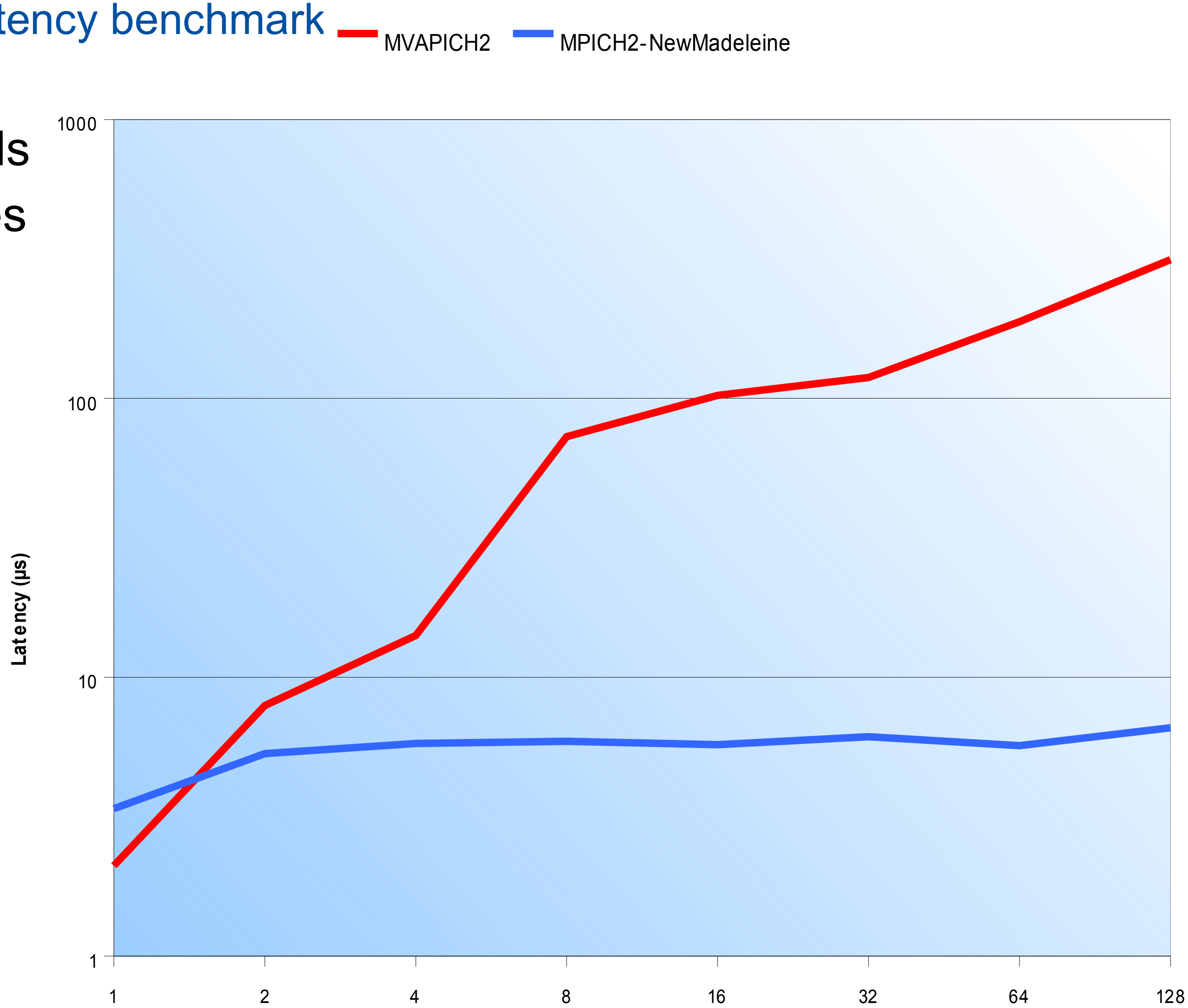




Multi-threaded latency
over Infiniband

- OMB multi-threaded latency benchmark
  - 1 sender thread
  - N receiver threads
  - 4-bytes messages

- MVAPICH2 1.2-p1
  - Active waiting
  - Concurrent polling
- OpenMPI 1.3.1
  - Segmentation fault
- NewMadeleine
  - Fixed-spin waiting
  - No concurrent polling
  - No overloaded CPU



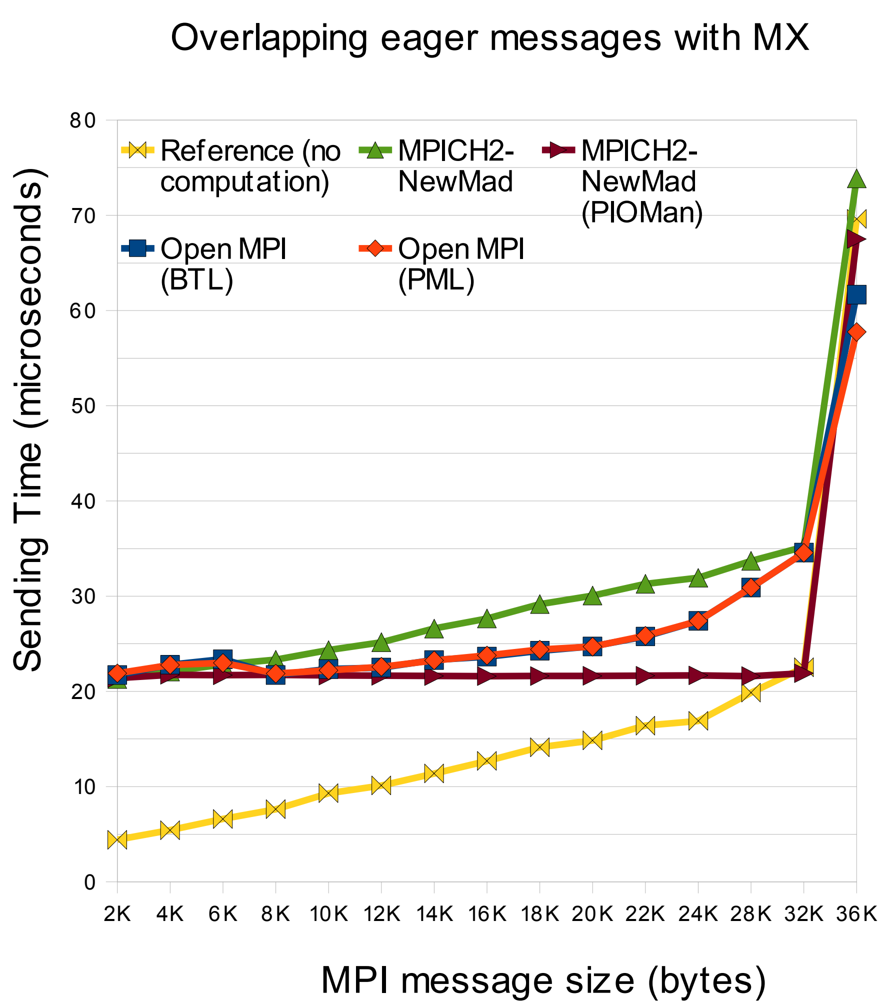
MPI overlap benchmark

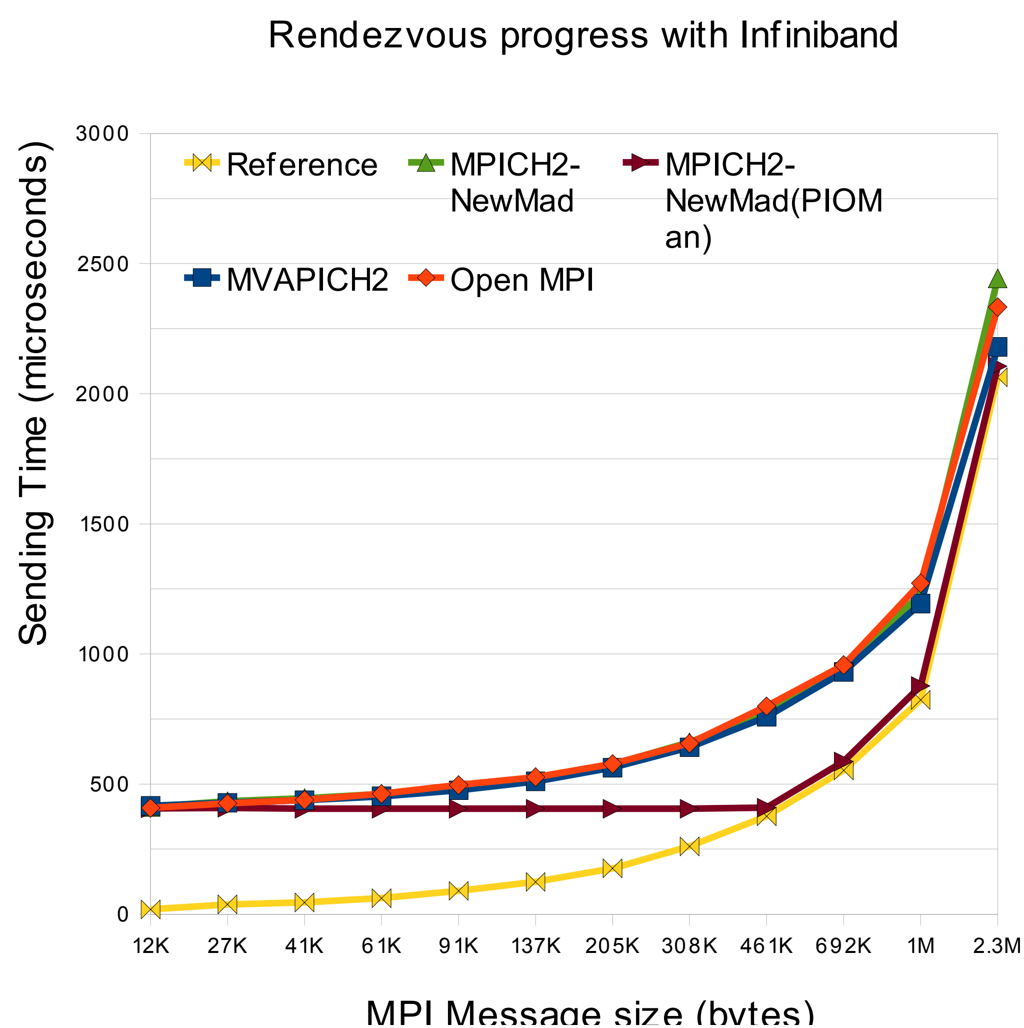
Time Measured

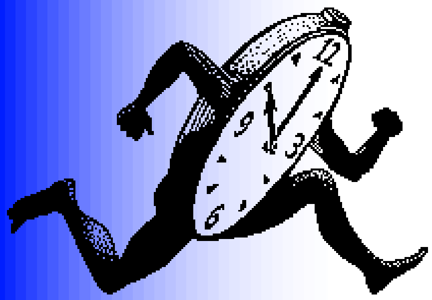
```
MPI_Isend(dest,sreq);
Computation();
MPI_Wait(&sreq);
MPI_Recv(dest);
```

- Computation time:
  - 20μs for small messages (*eager)*
  - 400μs for large messages (*rendezvous)*

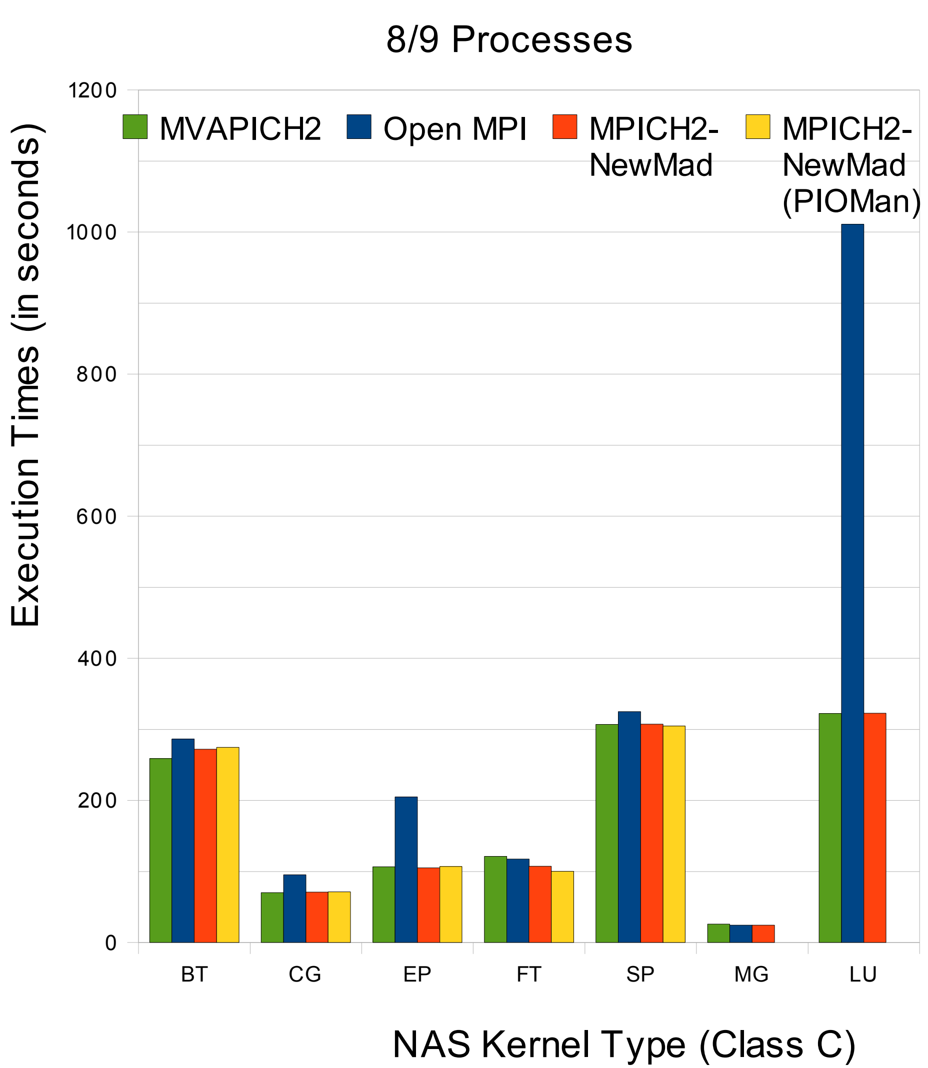
Communication progress with PIOMan

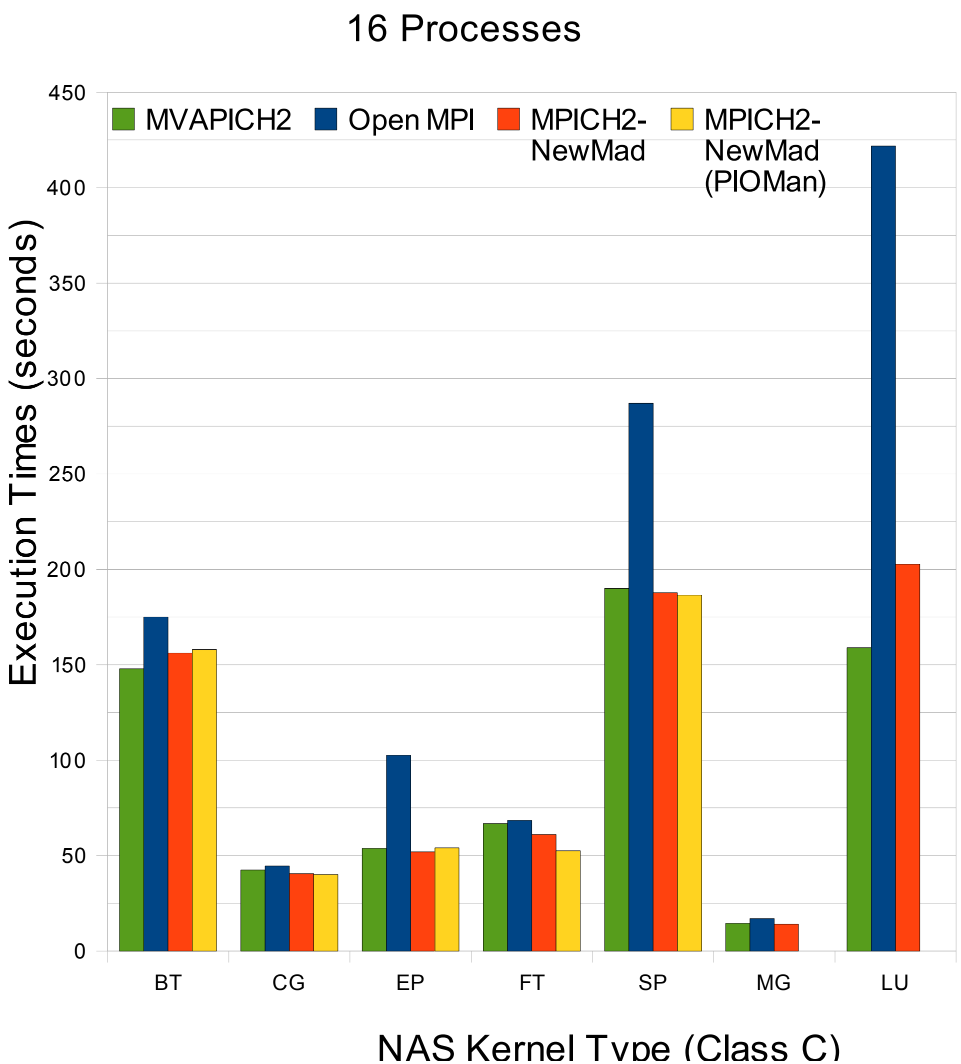




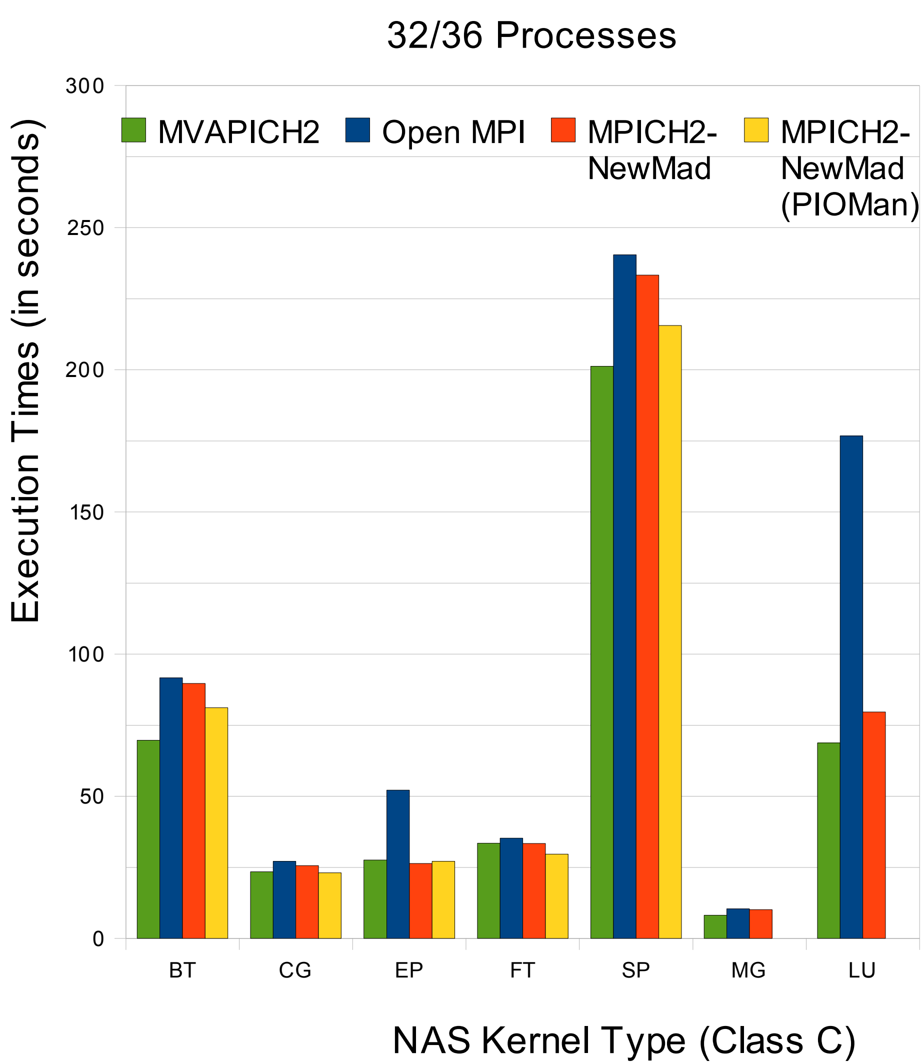


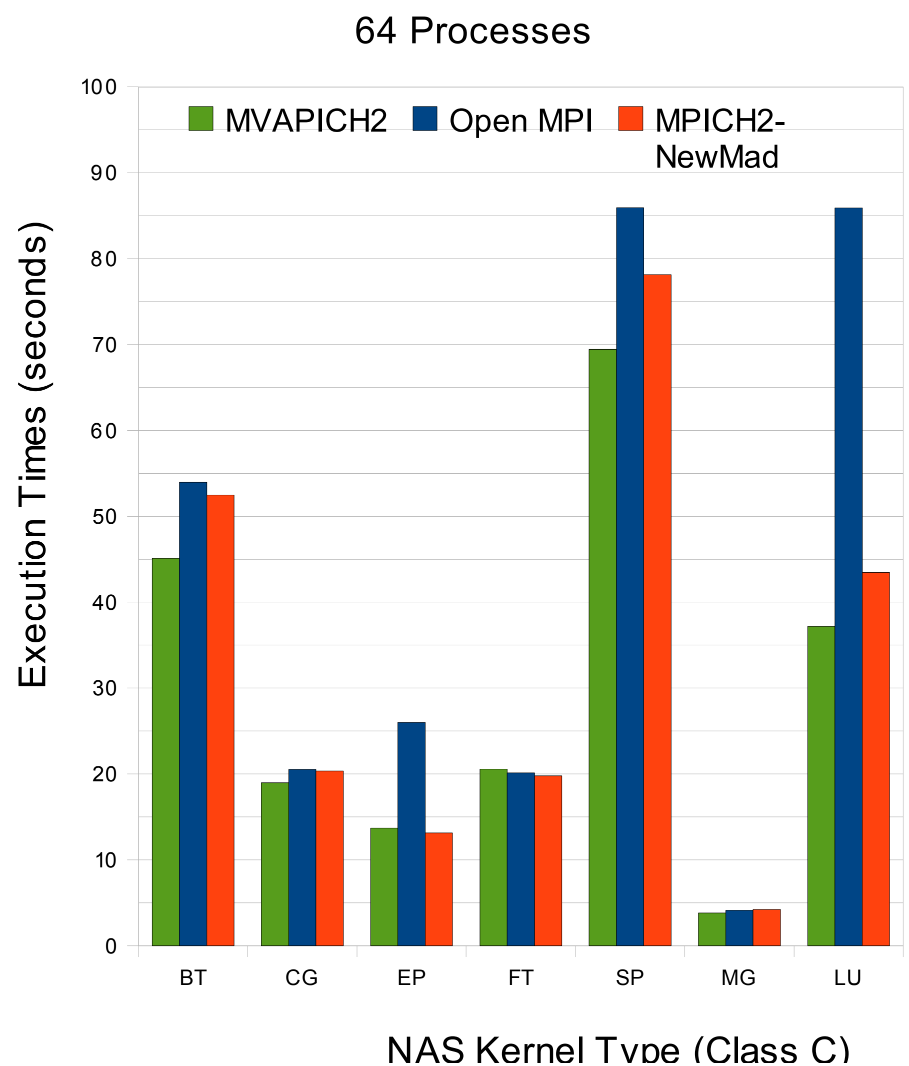
NAS Parallel Benchmarks





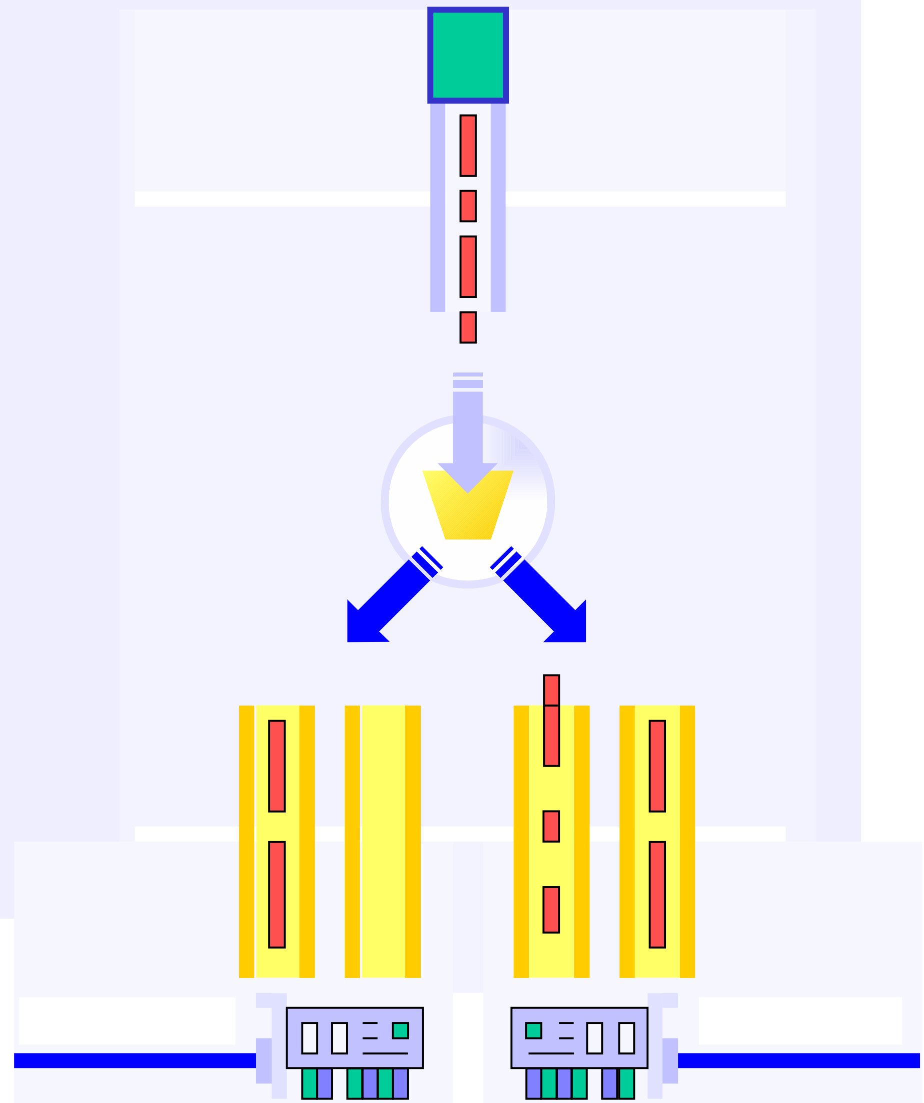
NAS Parallel Benchmarks (contd)





Heterogeneous multi-rail

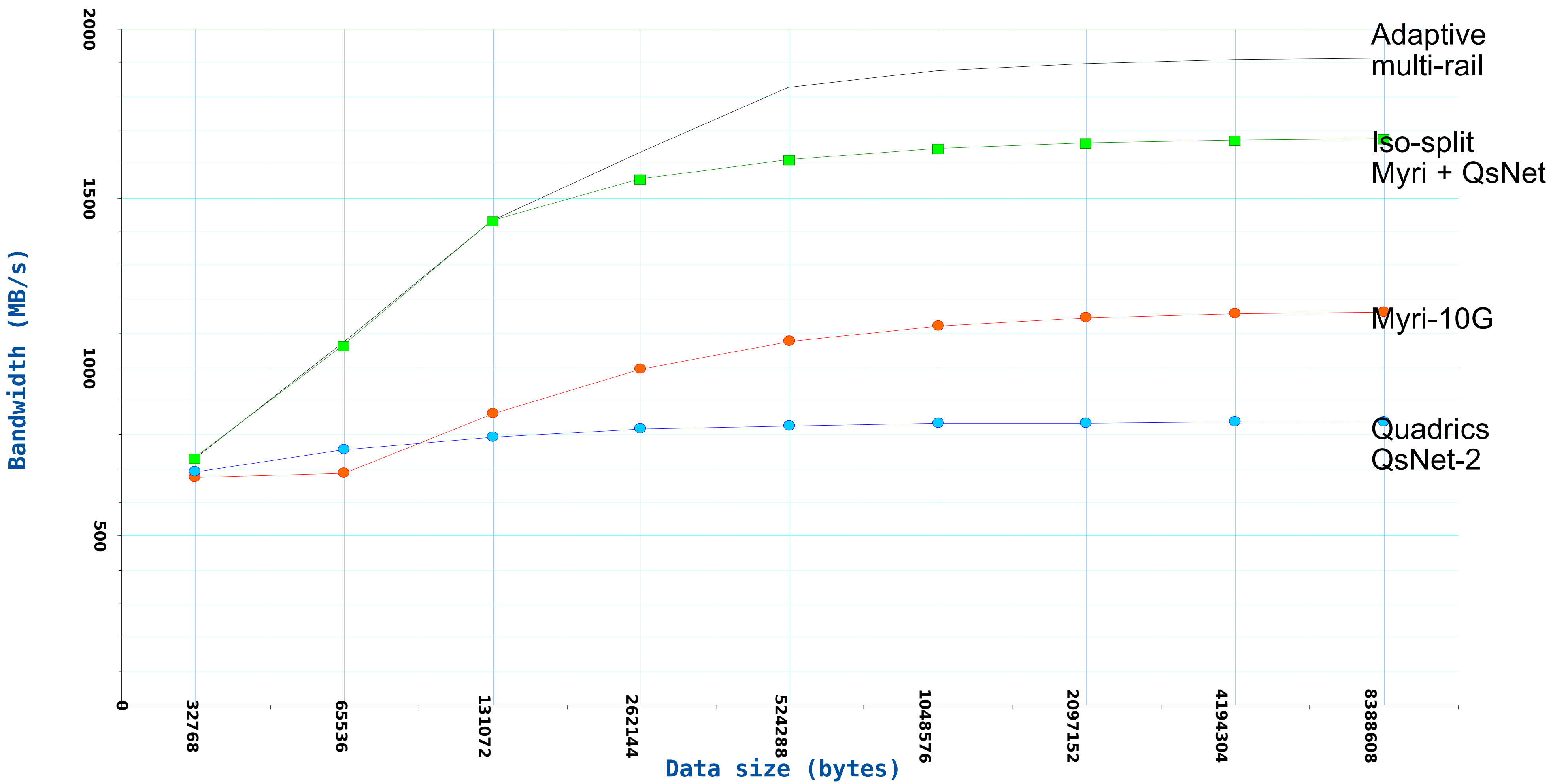
- Long fragments
  - “Heterogeneous” splitting
  - Split ratio computed according to:
    - Performance of each network
    - Estimated availability of NICs
  - Network sampling is necessary
- Short fragments
  - Aggregation over the fastest NIC



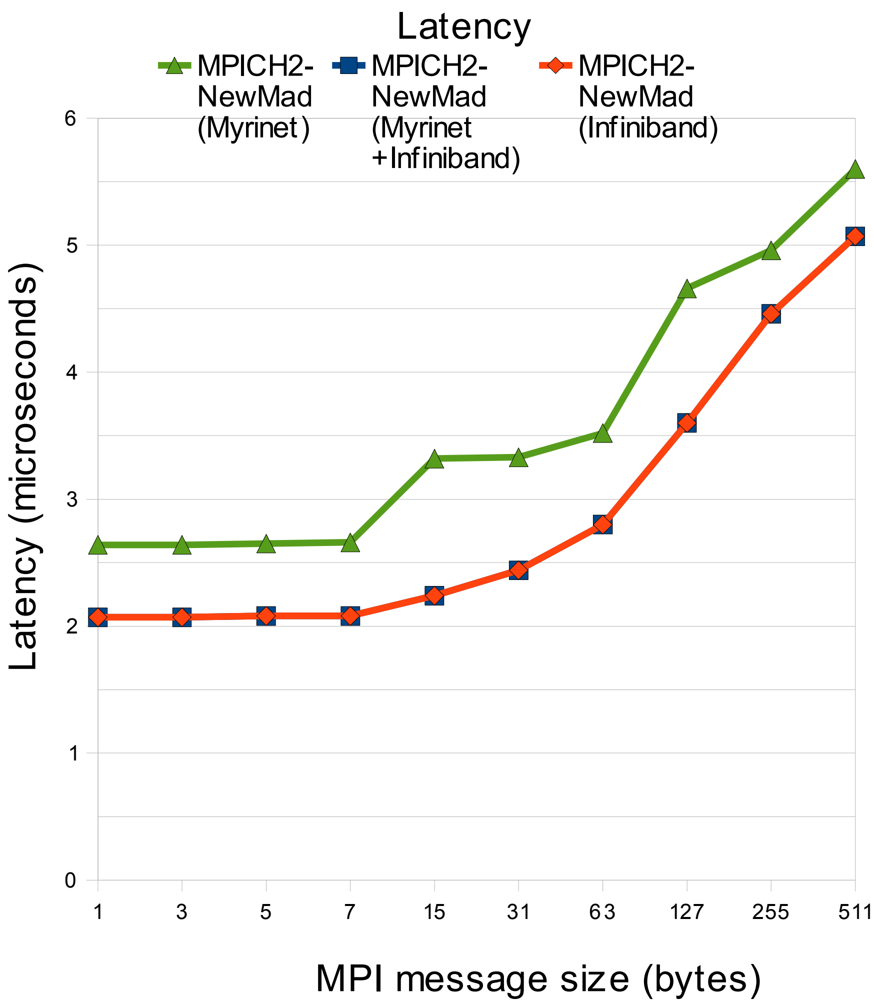
Aggregating bandwidth

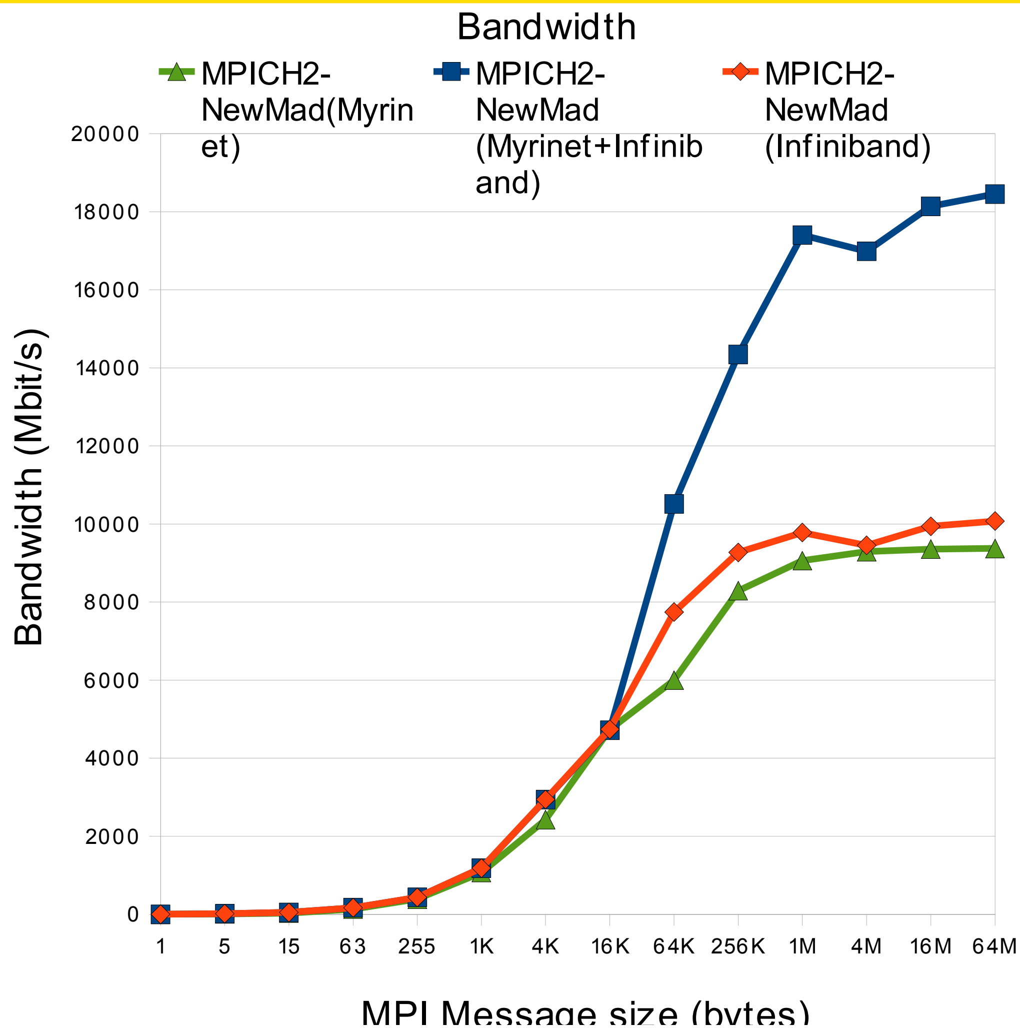
through adaptive multi-rail

NewMadeleine adaptive multi-rail: Myri-10G + QsNet-2



Point-to-point performance Adaptive Multirail



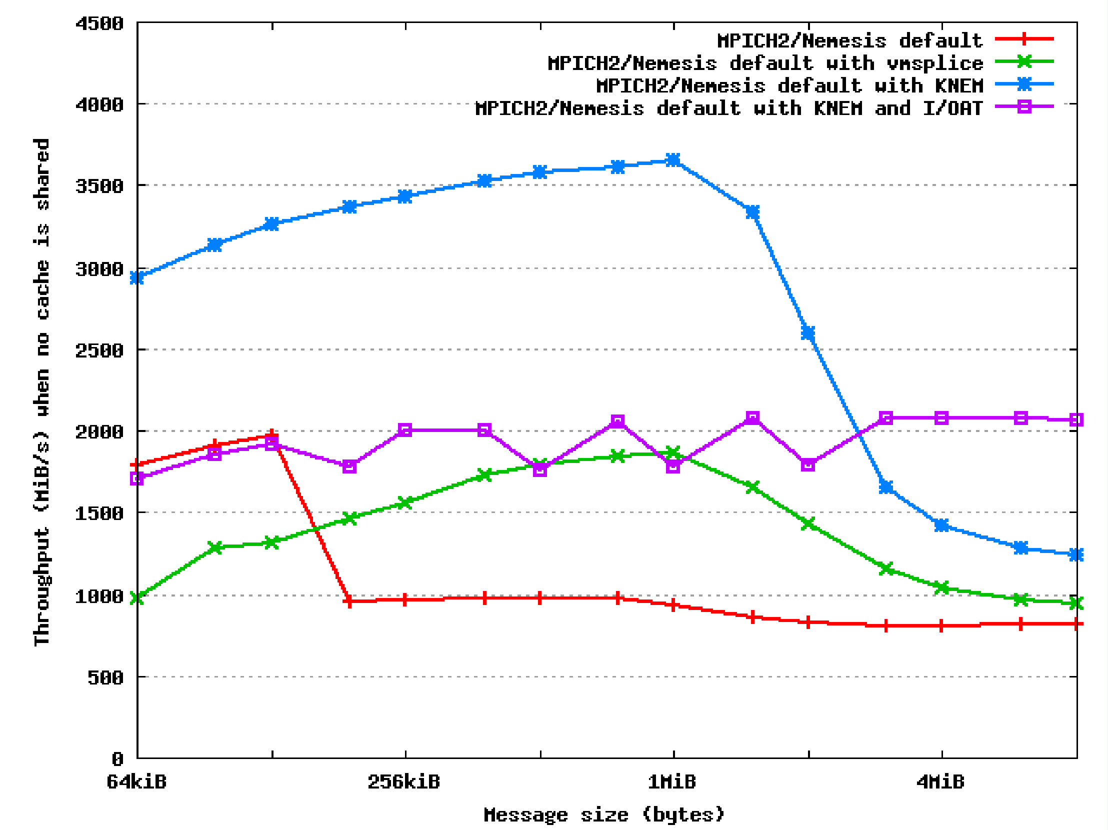


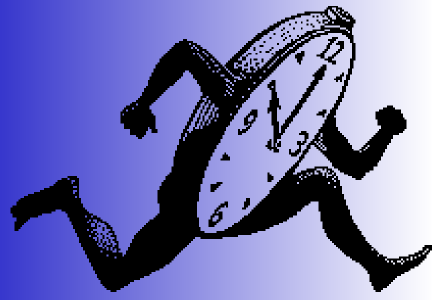
MPICH2-NewMad, point-to-point, Myrinet 10G NIC + ConnectX Infiniband HCA

High-throughput intra-node communications

- Increasing number of cores requires efficient intra-node communication
  - Existing strategy: double buffering
    - Consumes CPU cycles
    - Pollutes cache
- KNEM offers cheap alternative for large messages
  - Linux kernel-assisted memory data transfers
  - Support for non-contiguous/asynchronous transfers and for I/O AT copy offload
  - New backend in MPI2-Nemesis
  - Improves pt-to-pt and collectives performance significantly
    - Especially when no cache is shared
  - Developped in collaboration with ANL
    - Results to appear in ICPP (Vienna, sep 2009)

High-throughput intra-node communications





Conclusion

- The world is going multicore!
  - Massively multicore clusters may arrive sooner than expected
  - It has an impact on communication subsystem
- Mixing MPI and threads does not work out of the box
- Multicore is an opportunity to optimize communications
  - Optimization strategies of multiple flows
  - Use idle cores for communication progression

➔ We need new communication engines

- Designed from the groud up with multicore/multithread in mind
- Fully parallel
- NUMA-aware
- Machine-wide optimization of traffic

Thank you!

- More information:

http://runtime.bordeaux.inria.fr/

- Software available on INRIA Gforge:

http://gforge.inria.fr/projects/pm2/